UNIVERSIDADE FEDERAL DE VIÇOSA
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

RAFAEL ROSADO CRUZ

USO DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE SITUAÇÕES DE JOGO EM COMPETIÇÕES DE FUTEBOL DE ROBÔS

VIÇOSA
2012

RAFAEL ROSADO CRUZ

USO DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE SITUAÇÕES DE JOGO EM COMPETIÇÕES DE FUTEBOL DE ROBÔS

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 - Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.
Orientador: Prof. M.Sc. Alexandre S. Brandão.

VIÇOSA
2012

RAFAEL ROSADO CRUZ

USO DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE SITUAÇÕES DE JOGO EM COMPETIÇÕES DE FUTEBOL DE ROBÔS

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 - Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Aprovada em 05 de novembro de 2012.

COMISSÃO EXAMINADORA

Prof. M.Sc. Alexandre Santos Brandão - Orientador
Universidade Federal de Viçosa

Prof. Dra. Kétia Soares Moreira - Membro
Universidade Federal de Minas Gerais

Eng. B.Sc. Claudio Dario Rosales - Membro
Universidad Nacional de San Juan (Argentina)

Agradecimentos

Aproveito dessa seção para agradecer as instituições que confiaram e tornam possível a realização desse trabalho provendo fomento técnico e financeiro. Ao Departamento de Engenharia Elétrica da Universidade Federal de Viçosa (DEL/UFV), ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), a Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG), a Fundação de Apoio à Universidade Federal de Viçosa (FUNARBE) e à unidade de Viçosa da empresa UPTIME.

Rafael Rosado Cruz

"Ah! Se o mundo inteiro me pudesse ouvir. Tenho tanto pra contar, dizer que aprendi..."

Tim Maia

Resumo

A robótica é uma área do conhecimento abrangente, que engloba conceitos de diversas outras áreas do conhecimento, como computação, eletrônica, controle de sistemas, processamento de sinais, dentre outras. Dada sua vasta aplicação tecnológica, como em ambientes industriais ou hospitalares, é natural que hoje são desenvolvidas diversas pesquisas relacionadas a esta área. Uma das tendências, no entanto, é referente ao ramo da robótica móvel, pois acredita-se que a mesma trará valorosas melhorias para a qualidade de vida e produtividade do homem. Na Universidade Federal de Viçosa, a Equipe BDP/UFV conseguiu fundar um grupo atrativo que conciliasse desenvolvimento de pesquisa e qualificação pessoal neste segmento, com o entretenimento, por meio de uma equipe de futebol de robôs autônomos. A equipe atualmente trabalha com cinco robôs móveis que atuam de forma cooperativa e autônoma, baseada em programação prévia, em competições de futebol de robôs na categoria F-180, conforme as regras definidas pela Federação Internacional de Futebol de Robôs, FIRA. O sucesso das ações está diretamente relacionado ao posicionamento dos mesmos durante o jogo e, portanto, também está diretamente relacionada às estratégias de posicionamento e orientação em instantes presentes e futuros ao decorrer da partida. Entretanto, tais estratégias não devem ser imutáveis, pois devem estar adequadas as diferentes situações de jogo. Para que seja escolhida a estratégia mais pertinente, a equipe necessita de um método de classificação das situações do jogo. Tal método é assunto principal deste trabalho. As árvores de decisão são uma opção plausível para este tipo de problema, visto que pertencem a uma área da Inteligência Artificial que permite representar o conhecimento por meio de análises mais simples dos estados de um sistema. A construção de uma árvore de decisão inclui uma etapa denomina de aprendizagem, onde o projetista fornece um conjunto de exemplos de estado como entrada, além de uma resposta ou classe referente a aqueles exemplos. Daí aplica-se algum algoritmo que construa sequência de análises simples sobre os parâmetros de entrada e responda corretamente ao grupo de exemplos. Se o conjunto de exemplos for suficientemente grande e abrangente nas possibilidades de estado dos parâmetros de entrada, generaliza-se que o modelo classifica corretamente para quaisquer estados. Portanto esta ferramenta é uma opção poderosa e eficiente para classificar as situações de jogo e a atualização do algoritmo pode ser feita de forma dinâmica, inserindo exemplos retirados diretamente de um jogo corrente. A validação deste sistema pode ser verificada através da plataforma de simulação desenvolvida pela Equipe BDP/UFV. Os resultados obtidos pelas árvores de decisão podadas pelos algoritmo de Gini, Twoing e Deviance superaram percentualmente as classificações de situações indeterminadas de uma partida, quando comparada a regras heurísticas de determinação de um estado do jogo. Mais do que isto, foi possível observar que à medida que o banco de amostras cresce, melhor ocorre a classificação das árvores. Portanto, conclui-se que as árvores de decisão é uma ferramenta útil para a classificação de estados de jogo em uma competição de futebol de robôs autônomos.

Sumário

Lista de Figuras

1 Introdução

Até o momento, a maioria dos robôs instalados ao redor do mundo tem sido usados nas indústrias de processo de manufatura. Entretanto, o mundo em que vivemos tem mudado de formas sem precedentes à uma velocidade inimaginável. Embora a grande maioria dos robôs de hoje em dia ainda estarem nas fábricas, este avanço rápido da tecnologia tem permitido que os robôs passem a ser empregados na automação de várias tarefas em outros tipos de industria, como agrícolas, construção civil, saúde, entretenimento e educação.

Os robôs inteligentes, como sonhamos e vemos apenas em ficções científicas, novelas e filmes, estão se tornando realidade. Embora a robótica tenha uma história relativamente curta, em torno de algumas décadas, prevê-se que em breve os robôs pessoais entrarão em nossas casas e se tornarão parte muito importante do nosso dia-a-dia, assim como os celulares o são hoje em dia.

Porém, para que isso ocorra, dependemos de que várias pesquisas sejam desenvolvidas na robótica e em áreas relacionadas. Neste contexto, o futebol de robôs autônomos contribui para esse avanço, promovendo competições de tecnologia avançada no segmento da robótica móvel. As competições servem para que a nova geração de engenheiros e cientistas trabalhem com sistemas de robôs móveis autônomos em uma arena desafiadora, gerando conhecimentos que podem ser aproveitados em outras áreas.

É possível comprovar que trabalhar com futebol de robôs de fato implica em contribuições científicas. Para isso, basta observarmos as publicações relacionadas a esse tema nos últimos anos. Como exemplo, temos em [1] o detalhamento do projeto mecânico de um robô móvel omnidirecional, em [2] os autores apresentam uma solução de baixo custo para o sistema de controle de motores de corrente contínua, em [3] é apresentada uma estratégia de segmentação de imagens por cor utilizando redes neurais artificiais, em [4] temos o planejamento de trajetória para robôs móveis contemplando desvio de obstáculos utilizando de algoritmo genético e em [5] onde é feito a predição de movimento de robôs móveis em alta velocidade em um ambiente dinâmico utilizando redes neurais artificiais.

Ainda que não explorado no contexto de futebol de robôs, as Árvores de Decisão (ou Árvores de Classificação e Regressão) têm se mostrado uma ferramenta interessante vista sua vasta aplicação nas mais diferentes áreas. Por essa razão, podemos encontrar diversos artigos atuais utilizando dessa ferramenta, como em [6] onde o uso é feito para classificar a intensidade que as sílabas devem ser pronunciadas para melhorar a síntese digital de texto em fala, em [7] faz-se a classificação de usuários de uma rede social mais propensos a responder a uma campanha publicitária e em [8] onde é feito predição utilizando árvores de decisão e redes neurais artificiais da sobrevivência de de pacientes diagnosticados com cancer no fígado.

Este trabalho visa utilizar as Árvores de Decisão para classificar situações de jogo em partidas de futebol de robôs autônomos da liga *Small Size* na categoria *F-180*, para auxiliar as tomadas de decisão da equipe de futebol de robôs BDP da Universidade Federal de Viçosa.

1.1 Objetivos

O objetivo geral desse projeto foi desenvolver um sistema de classificação por meio de árvore de decisão capaz de classificar situações de jogo que até então eram indeterminadas no decorrer de uma partida de futebol de robôs. Como objetos específicos temos:

- Adequação da plataforma para reprodução de gravações de jogos com base nos registros do ssl-vision (small size league vision);
- Desenvolvimento um algoritmo para coleta dos exemplos para aplicação *on-line*;
- Construção dinâmica de uma árvore de classificação durante o jogo;
- Análise do desempenho do sistema na tarefa de classificar corretamente.

1.2 Estrutura do Projeto

Este projeto está organizado da seguinte forma:

- **Capítulo 1: Introdução**
  Este capítulo tem por finalidade apresentar uma breve revisão bibliografia, situando o leitor ao contexto sobre o qual essa pesquisa foi realizada. Além disso, há o propósito de explicitar o tema do projeto bem como as motivações que levaram a realização

do mesmo. Além disso, são apresentados os objetivos que pretende-se alcançar com a realização deste trabalho. Com esse propósito, neste capítulo o leitor encontrará informações sobre a plataforma de simulação sobre a qual os experimentos foram realizados, ressaltando suas principais características, a origem e natureza dos dados estudados no projeto e, de forma sucinta, a fundamentação teórica dos algoritmos de aprendizado de máquina utilizados neste projeto, suas origens e suas principais características, como aplicações, vantagens e limitações. Para esse último, também é apresentado ao leitor onde é possível encontrar a discussão desses temas de forma mais aprofundada.

- **Capítulo 2: Algoritmo de Extração de Exemplos**
  A finalidade do algoritmo de extração de exemplos é realizar a extração em tempo de jogo dos pares (x_i, y_i) onde x_i é vetor de estado de entrada ($x_i \in \mathbb{R}^n$) e y_i a classe ($y_i \in \{DefensivoComPosse, DefensivoSemPosse, OfensivoComPosse, OfensivoSemPosse\}$) para cada amostra i extraída. O propósito desse capítulo é explicar como é realizado esse procedimento.

- **Capítulo 3: Resultados e Discussões**
  Neste capítulo serão apresentados os resultados de performance das árvores de decisão como classificadores de situações de jogo. Há um comparativo do algoritmo proposto com um método heurístico, que também encontra-se explicado no capítulo, e por fim, é feita uma discussão acerca dos resultados obtidos.

- **Capítulo 4: Conclusões**
  Esse capítulo foi reservado para apresentar as conclusões a que se chegou com a realização desse trabalho, bem como considerações finais pertinentes.

1.3 A Plataforma BDP

A plataforma da equipe de futebol de robôs BDP/UFV, apresentada na Figura 1, foi construída em vista da necessidade por parte dos desenvolvedores de software de um ambiente gráfico que auxiliasse a análise quantitativa e qualitativa das técnicas e algoritmos.

Apesar do software ter sido desenvolvido nos paradigmas da programação estruturada, foram criadas estruturas e funções base que representam as principais características (como a posição dos robôs) e realizam as principais operações (como representar gráficamente os robôs).

Esse conjunto oferece a antigos e novos desenvolvedores da equipe um *framework* poderoso e eficaz no desenvolvimento de novos algoritmos, uma vez que exclui a necessidade do programador de saber como as principais operações funcionam na plataforma, bastando conhecer, se necessário, o que elas são destinas a executar.

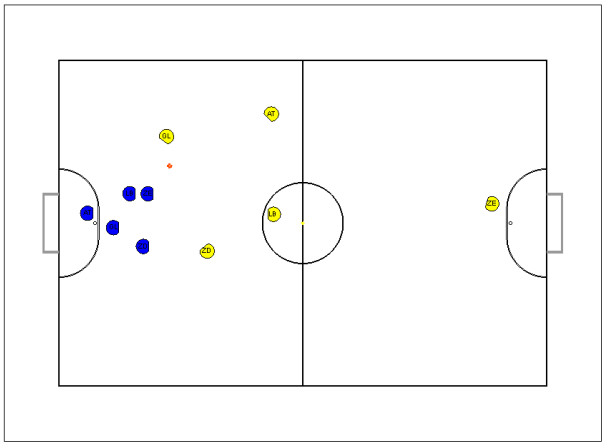


Figura 1: Plataforma da equipe de futebol de robôs BDP/UFV.

De fato, qualquer iniciante com conhecimentos básicos de programação na linguagem MATLAB, na qual a plataforma foi desenvolvidas, é capaz de incorporar um algoritmo próprio a plataforma, bastando conhecer os atributos principais, a tarefa das funções e as ordens de execução.

Além de agregar essa potencialidade de incorporação de novos códigos, a plataforma oferece um aumento de produtividade de seus utilizadores, facilitando a visualizar se os seus algoritmos operaram da forma como deveriam e o impacto deles no sistema geral. Desta forma, portanto, é possível através de simulações computacionais corrigir, aperfeiçoar e avaliar novos algoritmos.

1.4 O Banco de Dados

Entre 29 de julho e 05 de julho de 2009, aconteceu uma competição mundial de robôs inteligentes em Granz, Áustria, a RoboCup 2009. Com cerca de 3.000 participantes de 40 países foram disputadas diversas modalidades. Entre elas, a categoria *Small Size League F-180*, da qual a equipe BDP de futebol de robôs participa.

Afim de permitir que outras equipes possam reproduzir os jogos em softwares de simulação e realizar análises sobre o desempenho tático das equipes, ou até mesmo propor novos métodos, como é o caso deste trabalho, um dos participantes, Ulfert Nehmiz da

equipe B-Smart, registrou os dados de várias partidas proveniente do software *SSL-Vision (Small Size League Vision)*, que é responsável por realizar o processamento digital das imagens e enviar as informações dos estados da partida para os competidores.

Esses registos foram então utilizadas para alimentar as informações na plataforma de BDP, permitindo simular jogos reais.

1.5 Árvores de Decisão

Árvores de Classificação e Regressão consiste em um método de classificação que utilizada do histórico dos dados para construir o que é comumente chamado de Árvores de Decisão. As árvores de decisão são então utilizadas para a classificação de novos dados. Uma característica importante deste método é que o número de classes deve ser conhecido a priori.

A metodologia CART (do inglês: *Classification and Regression Trees*) foi desenvolvida na década de 80 por Breiman, Freidman, Olshen e Stone no paper "Classification and Regression Trees"(1984). Para construir uma árvore de decisão, o algoritmo CART utiliza o que é chamado de amostras de aprendizado - um conjunto de dados passados com classes conhecidas para todas observações. Por exemplo, amostras de aprendizado para um sistema de concessão de crédito deve fundamentalmente levar em conta informações sobre empréstimos passados (variáveis) relacionadas como resultado os pagamentos recentes (classes).

As árvores de decisão são representadas por um conjunto de questões que dividem as amostras de aprendizado em partes sucessivas, cada vez menores e mais refinadas. No algoritmo CART, essas questões são sempre perguntas binárias. Por exemplo, possíveis perguntas seriam: "A idade é maior que 50 anos?"ou "o sexo é masculino?". Portanto, fica a cargo do algoritmo olhar para todas as variáveis a disposição e todos os possíveis valores para tais variáveis, de modo a buscar as questões que subdividem o conjunto de amostras de aprendizado em duas partes, com o máximo de homogeneidade nas mesmas.

Uma vez realizado o processo e uma vez encontrada a questão que melhor desempenhe a tarefa de realizar a divisão de dois grupos homogêneos, o processo é repetido para cada novo grupo resultante da fragmentação, sucessivamente.

A compreensão do resultado desse algoritmo fica muito mais simples através de uma ilustração. Portanto, na Figura 2 esta um exemplo de uma árvore de classificação simples,

utilizada pelo *San Diego Medical Center* para classificar os pacientes em diferentes nível de risco.

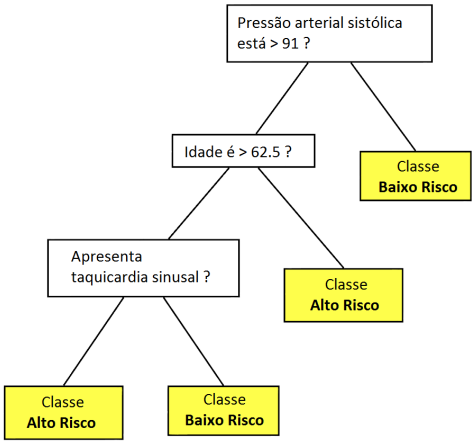


Figura 2: Árvore de classificação dos pacientes do San Diego Medical Center.

Na prática, surgem árvores de decisão muito mais complicadas do que a da Figura 2, contendo dezenas de camadas e centenas e variáveis. A Figura 2 também nos mostra que o algoritmo é capaz de lidar fácilmente tanto com variáveis numéricas quanto com variáveis categóricas.

Entre outras vantagens, o método CART é robusto contra dados discrepantes. Normalmente o algoritmo que faz as divisões do conjunto de dados isola os dados discrepantes em nodos individuais.

Outra importante propriedade prática do algoritmo é que a estrutura da árvore de classificação ou regressão é invariante perante transformações monótonas em variáveis independentes, ou seja, pode-se substituir qualquer variável pelo seu valor da raiz quadrada ou logaritmo que a estrutura da árvore não modificará.

Podemos resumir o algoritmo em três partes:

1. Construir a árvore completa (*maximum tree*).
2. Determinar o tamanho correto da árvore.
3. Fazer a classificação dos novos dados utilizando a árvore obtida.

Como o propósito desse trabalho é aplicar o algoritmo CART para a construção de uma Árvore de Classificação, as etapas acima serão discutidas a seguir levando em consideração este contexto. Se for do interesse do leitor, todas as informações sobre o algoritmo como um todo podem ser encontradas de forma bem clara e condensada em [9].

1.5.1 Árvore completa

A determinação de uma árvore completa é a parte que consome mais tempo do projetista. Construir a árvore completa (*maximum tree*) implica em particionar as amostras de aprendizado até uma última observação, ou seja, até que o último subgrupo formado seja indivisível por só conter uma amostra ou os nodos terminais contém observações de apenas uma classe.

Neste momento, é importante recordarmos que as árvores de classificação são utilizadas quando a classe é conhecida a priori para cada observação do conjunto de amostras de aprendizado. Essas classes comumente são determinadas pelo próprio conhecimento do projetista.

Para fim de ilustração do processo, devemos considerar nessa seção t_p como um nodo pai, e t_e, t_d - respectivamente nodos filhos esquerdo e direito de t_p. Além disso, condiremos a matriz X com o conjunto de amostras de aprendizado, com M variáveis x_j e N observações, e um vetor Y contendo a classe para as N observações, sendo que as mesmas são classificadas em uma das K possíveis classes.

As árvores de classificação são construidas a partir da regra de divisão (*splitting rule*) - a regra que realiza a divisão do conjunto de aprendizado em partes grupos menores. Como já discutido, a cada passo os dados são divididos em dois grupos com o máximo de homogeneidade nos dados de cada grupo.

A máxima homogeneidade dos nodos filhos é definida a partir do que é chamado de função de impureza (*impurity function*) $i(t)$. Uma vez que a impureza de um nodo pai t_p é constante para qualquer divisão possível $x_j \leq x_j^R, j = 1, ..., M,$, a máxima homogeneidade dos nodos filhos esquerdo e direito serão equivalente a maximização da diferença na função de impureza $\Delta i(t)$:

$$\Delta i(t) = i(t_p) - P_e i(t_e) - P_d i(t_d) \tag{1.1}$$

onde na equação 1.1 P_e e P_d são as probabilidades para os nodos filhos esquerdo e direito respectivamente.

Em outras palavras, o algoritmo CART se resume a resolver para cada nodo o seguinte problema de maximização:

$$\underset{x_j \leq x_j^R, j=1,...,M}{\text{argmax}} \quad [i(t_p) - P_e i(t_e) - P_d i(t_d)] \tag{1.2}$$

A figura 3 ilustra bem o processo conforme descrito até aqui.

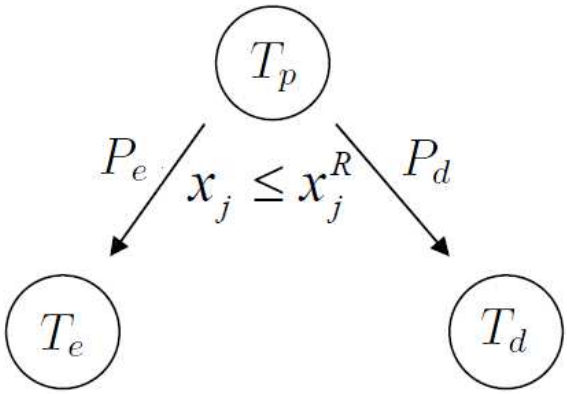


Figura 3: Algoritmo de divisão do CART, onde t_p, t_e, t_d - nodos pai, esquerdo e direito; x_j - variável j; x_j^R - melhor valor de divisão para a variável x_j.

A equação 1.2 implica que o algoritmo CART vai procurar através de todos os possíveis valores de todas as variáveis na matriz X pela melhor questão de divisão $x_j \leq x_j^R$ que irá maximizar a diferença na medida me impureza $\Delta i(t)$.

A próxima importante questão é como definir a função de impureza $i(t)$. Teoricamente há diversas funções de impureza, entretanto na prática apenas duas delas são largamente utilizadas: regra de divisão de Gini (*Gini splitting rule*) e regra de divisão de Twoing (*Twoing splitting rule*).

1.5.1.1 Regra de divisão de Gini

A regra de divisão de Gini (ou indexação de Gini) é a mais utilizada. Ela utiliza, de forma genérica, a seguinte função de impureza $i(t)$:

$$i(t) = \sum_{k \neq l} p(k|t)p(l|t) \tag{1.3}$$

Onde na equação 1.3 k e l iguais a $1, ..., K$ - correspondendo ao índice das possíveis classes; $p(k|t)$ - a probabilidade condicional da classe k assumindo que estamos no nodo t.

Aplicando a função de impureza de Gini (1.3) ao problema de maximização (1.2), nós

teremos a seguinte variação na medida de impureza $\Delta i(t)$:

$$\Delta i(t) = -\sum_{k=1}^{K} p^2(k|t_p) + P_e \sum_{k=1}^{K} p^2(k|t_e) + P_d \sum_{k=1}^{K} p^2(k|t_d)$$

Desta forma, portanto, a utilização do algoritmo Gini acarreta na resolução do seguinte problema:

$$\underset{x_j \leq x_j^R, j=1,\ldots,M}{\operatorname{argmax}} [-\sum_{k=1}^{K} p^2(k|t_p) + P_e \sum_{k=1}^{K} p^2(k|t_e) + P_d \sum_{k=1}^{K} p^2(k|t_d)] \tag{1.4}$$

Observe da equação 1.4 que o algoritmo irá buscar no conjunto de amostras de aprendizado pela maior classe e irá isola-la do restante dos dados. Este algoritmo funciona bem ao lidar com dados ruidosos.

1.5.1.2 Regra de divisão de Twoing

Ao contrário da regra de divisão de Gini, a regra Twoing irá buscar por duas classes que juntas correspondam a mais de 50% dos dados, no conjunto de amostras de aprendizado. A regra de divisão de Twoing consiste na maximização para a seguinte variação na medida de impureza $\Delta i(t)$:

$$\Delta i(t) = \frac{P_e P_d}{4} \Big[\sum_{k=1}^{K} \mid p(k|t_e) - p(k|t_d) \mid\Big]^2$$

A utilização do algoritmo Twoing implica portanto no seguinte problema de maximização:

$$\underset{x_j \leq x_j^R, j=1,\ldots,M}{\operatorname{argmax}} \Big(\frac{P_e P_d}{4} \Big[\sum_{k=1}^{K} \mid p(k|t_e) - p(k|t_d) \mid\Big]^2\Big) \tag{1.5}$$

Apesar da regra de divisão de Twoing nos permitir construir árvores mais balanceadas, este algoritmo é mais lento que a regra de Gini. Para exemplificação, considerando o número total de classes igual a K, para cada iteração temos sempre 2^{K-1} possibilidades de divisão a serem analisadas.

Além das regras de divisão de Gini e Twoing mencionadas, existem diversos outros métodos. depois desses, os mais utilizados são a regra da Entropia (*Enthropy rule*), regra

χ^2 e regra do desvio máximo (*maximum deviation*). Mas em [9] os autores provam (página 95) que a árvore final independe da escolha da regra de divisão, porém afirmam que o processo de podagem (*pruning*) é muito mais importante.

1.5.2 Determinação do Tamanho Correto da Árvore

A árvore completa (*maximum tree*) pode se tornar muito complexa, consistindo de centenas de níveis. Portanto, a mesma deve ser otimizada antes de ser utilizada para a classificação de novos dados.

A otimização da árvore significa escolher o tamanho adequado, ou seja, remover os nodos insignificantes ou até mesmo sub-árvores. Para realizar esta tarefa, a qual se denomina podagem, existem dois algoritmos muito utilizados na prática: otimização por número de pontos em cada nodo, e validação cruzada.

1.5.2.1 Otimização pelo Número Mínimo de Pontos

Neste caso, fazemos com que a divisão seja interrompida quando o número de observações no nodo é menor do que um número requerido mínimo pré-definido N_{min}. Obviamente, quanto maior for o parâmetro N_{min}, menor será o tamanho da árvore.

Por um lado, esta abordagem apresenta as vantagens de ser um método muito rápido, é muito fácil de aplicar e apresenta resultados consistentes. Por outro lado, exige que seja feita a calibração de um novo parâmetro, o N_{min}. Na prática, é comum que se utilize N_{min} como sendo 10% do total de exemplos do conjunto de amostras de aprendizado.

Quando temos a tarefa de definir o tamanho da árvore, existe um problema de interesses conflitantes entre a medida de impureza e a complexidade da árvore, que é definida pelo número total de nodos terminais na árvore $\tilde{T}$. Isso se dá ao fato de que uma árvore menor apresenta menor custo computacional, enquanto uma árvore maior apresenta maior precisão na classificação.

De fato, na árvore completa temos que a medida de impureza será mínima e igual a zero, porém o número de nodos terminais $\tilde{T}$ será máximo. Para encontrar o valor ótimo da árvore, uma opção é utilizar a validação cruzada.

2 Algoritmo de Extração de Exemplos

O primeiro problema que surge quando se pretende iniciar a análise dos dados está relacionado a forma como eles foram capturados do *SSL-Vision*. Em vista do posicionamento das câmeras, as quais cada uma foi propositalmente posicionada de modo a cobrir pouco mais de meio campo, e a alta frequência de amostragem das mesmas, temos como resultado o envio multiplexado dos frames individuas de cada câmera, onde na maioria dos casos não há todos os robôs detectados em um mesmo frame.

Essa característica da montagem do sistema descrita acima resulta em um problema de atribuição, uma vez que o número total de robôs no campo é conhecido, mas a cada nova informação recebida deve-se tomar o cuidado de atualizar as informações dos robôs corretos. Esta etapa, apesar de preceder o início do algoritmo de extração dos dados, está explicada nessa seção. Por ser imprescindível na etapa de extração, se as atribuições forem feitas de forma incorreta, aparecerão inconsistências nos dados que podem comprometer todo o experimento.

Portanto, para que fossem escolhidos os robôs a ser atualizado a cada frame, utilizou-se uma análise da distância euclidiana entre a posição atual para cada um do conjunto total de robôs, e as novas posições detectadas. Os pares com menor distância euclidiana recebem a atualização, enquanto os outros robôs continuam na última posição detectada. Esse procedimento é repetido continuamente durante a simulação.

Agora, com o sistema preparado para realizar a coleta dos exemplos que alimentarão a árvore, inicia-se a formulação do problema. Apesar de estarmos trabalhando sobre uma gravação, sabendo que o sistema deve se adaptar ao decorrer do tempo em uma partida, faz-se necessário a implementação de um algoritmo de extração que possa ser realizado de forma *on-line* e *off-line*. Desta forma, considerando o fator temporal, as tarefas devem ser realizadas na seguinte sequencia (do ponto de vista de que pertencemos ao time azul):

1. Verifica-se o estado do Juiz até que a situação de jogo fique corrente.

2. Quando todos os robôs estiverem a uma distância da bola superior a δ, é considerado uma situação indeterminada, portanto é registrado o posicionamento e a velocidade dos robôs e da bola, para que possa ser utilizado como um possível exemplo.

3. Aguarda-se até que a distância entre algum dos robôs em relação a bola seja inferior a ϵ. Se antes disso o Juiz interromper o jogo, reinicia-se o procedimento.

4. Analisa-se a posição da bola: se estiver no meio de campo do time azul, é feita a marcação sobre o último registro de possível exemplo que trata-se de uma situação Defensiva, caso contrário é feita a marcação de situação Ofensiva.

5. Analisa-se qual foi o robô que primeiro se aproximou a uma distância ϵ da bola: se for do time azul, é feita a marcação sobre o último registro de possível exemplo que trata-se de uma situação "com posse de bola", caso contrário é feita a marcação "sem posse de bola".

6. Se o robô que primeiro se aproximou a uma distância ϵ da bola ficar por menos que cinco frames consecutivos com distância para a bola menor que ϵ, o procedimento é re-inicializado.

7. As marcações para o possível exemplo são combinadas para constituir uma das possíveis classes: "Defensiva sem posse de bola", "Defensiva com posse de bola", "Ofensiva sem posse de bola"ou "Ofensiva com posse de bola".

8. O registro do possível exemplo e sua classe referente são transferidos para um banco de dados permanente.

9. A árvore é re-construída com todos os exemplos do banco de dados permanente.

É importante frisar que uma classe é estabelecida quando há uma transição entre uma situação indeterminada e uma situação determinada. As possíveis classes que qualificam a situação indeterminada estão listadas no item 7.

Com base no dimensionamento dos objetos e após a realização de alguns experimentos, foi estipulado empíricamente os valores de $\delta = 0.3m$ e $\epsilon = 0.15m$.

As Figuras 4 e 5 ajudam na compreensão do algoritmo, ilustrando os momentos cruciais em que é detectado um possível exemplo, e o momento em seguinte em que é detectado a classe ao qual o exemplo pertence.

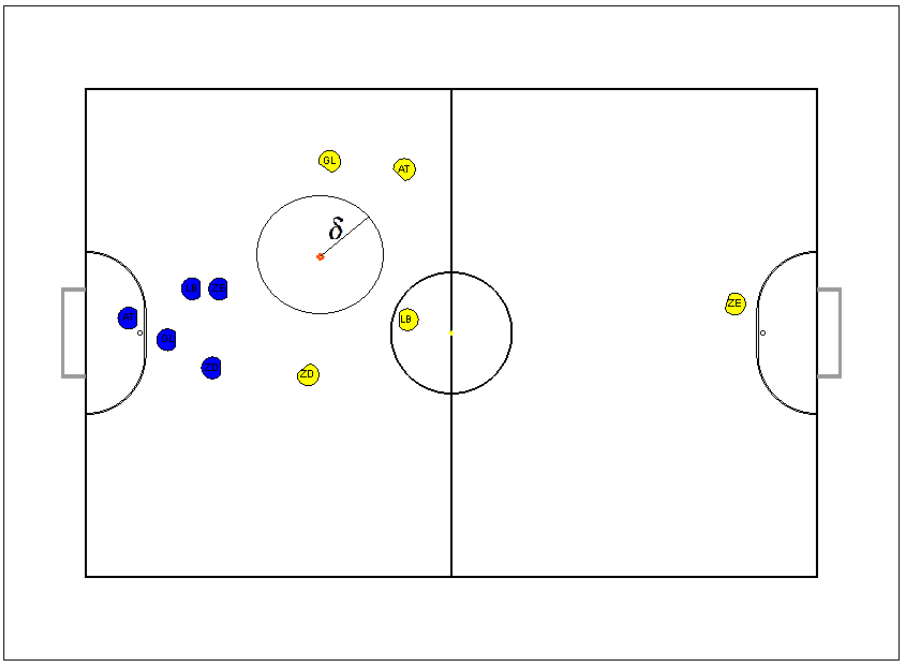


Figura 4: Momento em que é considerado uma situação indeterminada, e é registrado um possível exemplo a procura de classe.

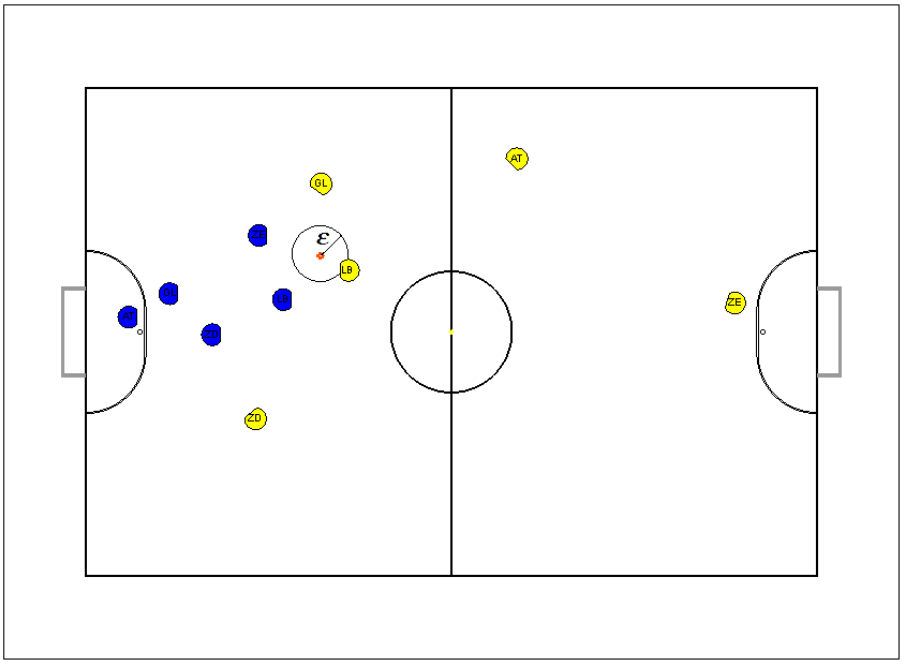
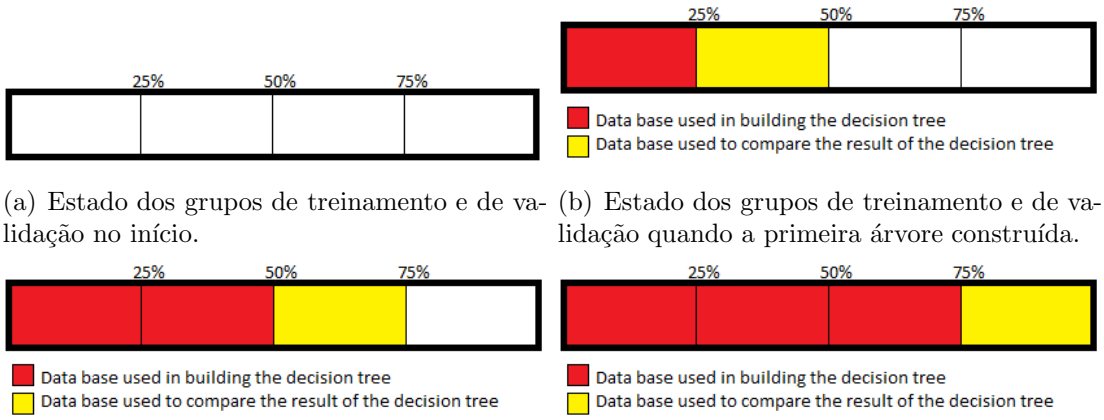


Figura 5: Momento em que a classe para o exemplo extraido representado na Figura 4 é determinada.

3 Resultados e Discussões

O algoritmo utilizado para construção das árvores é o algoritmo CART original, proposto por *Breiman, Freidman, et all* no artigo *"Classification and Regression Trees"(1984)* [9].

Este capítulo apresenta o desempenho das árvores de decisão quando são utilizados 25%, 50% e 75% do total de amostras obtidas durante uma partida para treinar o modelo. As divisões dos dados utilizados em treinamento e validação podem ser explicadas com mais clareza através das Figuras 6(a), 6(b), 6(c) and 6(d), onde estão representados os estados dos grupos de treinamento e de validação em relação ao conjunto total de amostras analisadas.



(a) Estado dos grupos de treinamento e de validação no início.

(b) Estado dos grupos de treinamento e de validação quando a primeira árvore construída.

(c) Estado dos grupos de treinamento e de validação quando a segunda árvore construída.

(d) Estado dos grupos de treinamento e de validação quando a terceira árvore construída.

Figura 6: Estado dos grupos de treinamento e de validação.

Para conferir os resultados, foram analisadas as seguintes partidas:

1. B-Smart x ER-Force
2. PlasmaZ x ZjuNlict
3. PlasmaZ x B-Smart

Nas Figuras 7, 8 and 9 estão representados o percentual de respostas corretas providas pela árvores em cada partida, na ordem em que foram enumeradas acima. Para a construção das árvores foram utilizados os algoritmos de divisão de Gini, Twoing e Deviance, como explicado em [9].

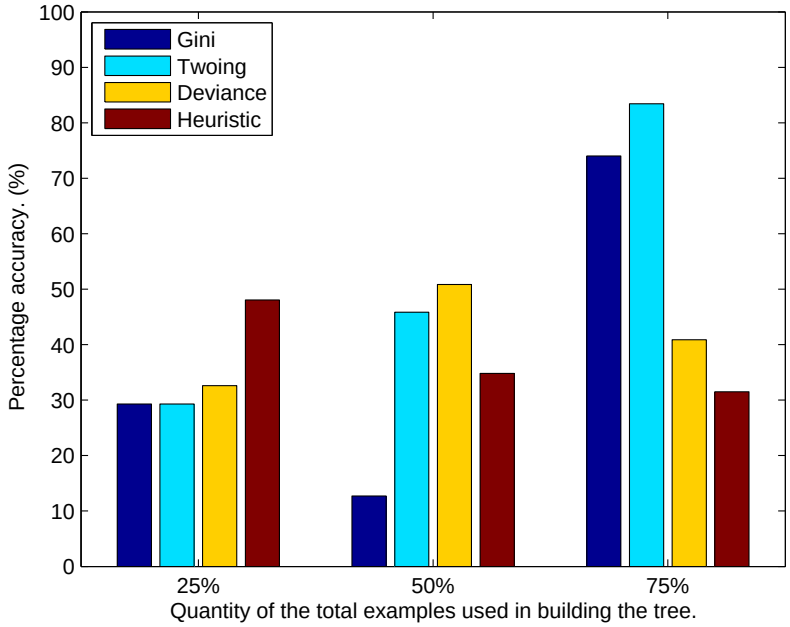


Figura 7: Performance dos algoritmos na partida entre B-Smart e ER-Force.

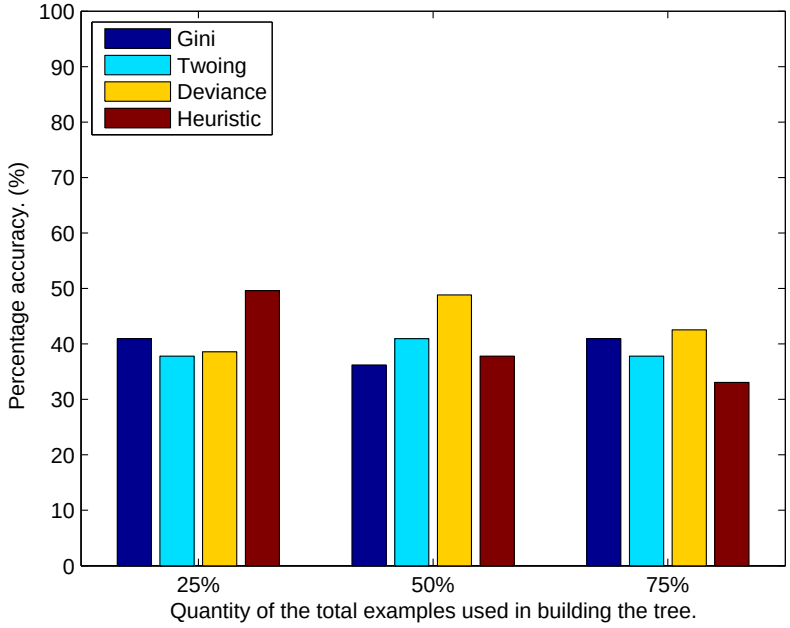


Figura 8: Performance dos algoritmos na partida entre PlasmaZ e ZjuNlict.

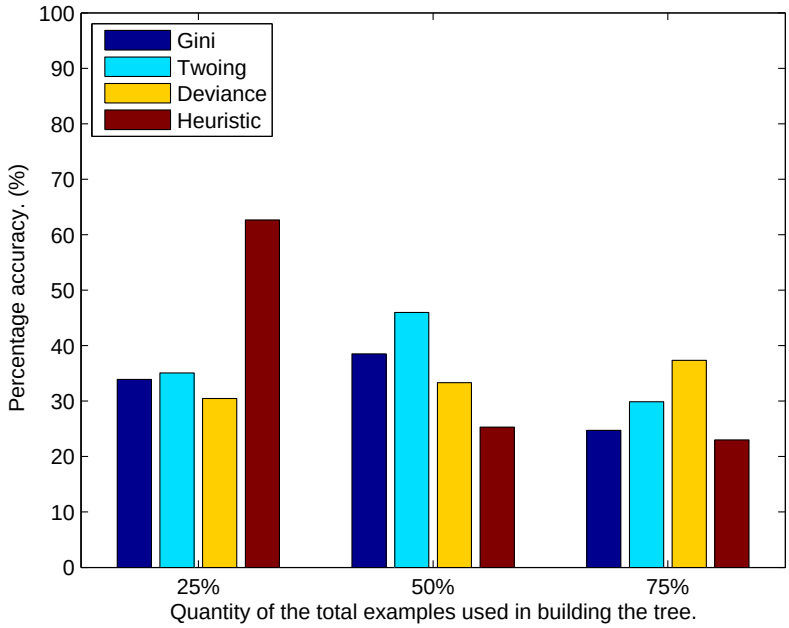


Figura 9: Performance dos algoritmos na partida entre PlasmaZ e B-Smart.

Para efeito de comparação, os resultados foram comparados com uma regra simples condicional que não faz uso da árvore. No gráfico, tal regra está com o nome de "Heurística", e os valores foram calculados da seguinte forma:

1. Para cada exemplo do banco de dados, verifica-se a posição da bola: se estiver no meio de campo do time azul, é feita a marcação sobre exemplo que trata-se de uma situação Defensiva, caso contrário é feita a marcação de situação Ofensiva.

2. Verifica-se qual o jogador mais próximo da bola: Se o jogador mais próximo percenter ao time azul, é feita a marcação sobre o exemplo que trata-se de uma situação "com posse de bola", caso contrário é feita a marcação de situação "sem posse de bola".

3. As marcações para exemplo são combinadas para constituir uma das possíveis classes: "Defensiva sem posse de bola", "Defensiva com posse de bola", "Ofensiva sem posse de bola"ou "Ofensiva com posse de bola".

4. Esta classe é associada ao exemplo, para o algoritmo "Heurística".

Através dos gráficos é possível observar que os resultados melhoram conforme se aumenta o conjunto de dados de treinamento, o que era obviamente o esperado, uma vez que uma grande quantidade de informação fornecida melhora a capacidade de generalização do modelo.

4 Conclusões

Hoje em dia existem diversos algoritmos sofisticados capazes de fazer o reconhecimento de padrões e classificação de dados, entretanto, as competições de futebol de robôs exigem que as decisões sejam tomadas rapidamente, por isso algoritmos de baixo custo computacional como as árvores de decisão se tornam atraentes para esse tipo de aplicação.

O algoritmo CART, em específico, aparentava promissor, visto a simplicidade de implementação e a vasta aplicabilidade. Como esperado, as árvores de decisão apresentaram melhor resultado do que a tomada de decisão baseada em heurística, uma vez que elas levam em consideração um número de amostras passadas para tomar a decisão futura.

Este trabalho discute uma aplicação desta ferramenta ressaltando os bons resultados. Como proposta de trabalho futuro, pretende-se analisar os parâmetros de entrada de treinamento da árvore. Neste sentido, pensa-se em levar em conta novos parâmetros, tais como o estado do juiz no tempo em que ocorreu uma situação indeterminada.

Referências

[1] NASCIMENTO, T. do; COSTA, A. da; PAIM, C. Axebot robot the mechanical design for an autonomous omnidirectional mobile robot. In: *Electronics, Robotics and Automotive Mechanics Conference, 2009. CERMA '09.* [S.l.: s.n.], 2009. p. 187 –192.

[2] XIANWEI, W.; KARNJANADECHA, M.; KHAORAPAPONG, T. A low-cost solution of motor control system for robocup robots. In: *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2008. ECTI-CON 2008. 5th International Conference on.* [S.l.: s.n.], 2008. v. 2, p. 645 –648.

[3] TORRES, E.; WEITZENFELD, A. Robocup small-size league: Using neural networks to learn color segmentation during visual processing. In: *Robotic Symposium, 2008. LARS '08. IEEE Latin American.* [S.l.: s.n.], 2008. p. 14 –19.

[4] BURCHARDT, H.; SALOMON, R. Implementation of path planning using genetic algorithms on mobile robots. In: *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on.* [S.l.: s.n.], 2006. p. 1831 –1836.

[5] SHENG, Y.; WU, Y. Motion prediction in a high-speed, dynamic environment. In: *Tools with Artificial Intelligence, 2005. ICTAI 05. 17th IEEE International Conference on.* [S.l.: s.n.], 2005. p. 3 pp. –705. ISSN 1082-3409.

[6] VEMPADA, R.; RAO, K. Modeling the intensity of syllables using classification and regression trees. In: *Communications (NCC), 2012 National Conference on.* [S.l.: s.n.], 2012. p. 1 –5.

[7] SURMA, J.; FURMANEK, A. Data mining in on-line social network for marketing response analysis. In: *Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom).* [S.l.: s.n.], 2011. p. 537 –540.

[8] CHEN, C.-M. et al. Prediction of survival in patients with liver cancer using artificial neural networks and classification and regression trees. In: *Natural Computation (ICNC), 2011 Seventh International Conference on.* [S.l.: s.n.], 2011. v. 2, p. 811 –815. ISSN 2157-9555.

[9] BREIMAN, L. et al. *Classification and Regression Trees.* New York: Chapman & Hall, 1984. 358 p. New edition of [10]. ISBN 0-412-04841-8. Disponível em: <http://www.crcpress.com/catalog/C4841.htm>.

[10] Breiman, L. et al. *Classification and Regression Trees.* Belmont, California, U.S.A.: Wadsworth Publishing Company, 1984. (Statistics/Probability Series).